Bencode

From Wikipedia, the free encyclopedia

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. *(September 2013)*

Bencode (pronounced like *B encode*) is the encoding used by the peer-to-peer file sharing system BitTorrent for storing and transmitting loosely structured data.

It supports four different types of values:

- byte strings,
- integers,
- lists, and
- dictionaries (associative arrays).

Bencoding is most commonly used in torrent files. These metadata files are simply bencoded dictionaries.

While less efficient than a pure binary encoding,[*citation needed*] bencoding is simple and (because numbers are encoded as text in decimal notation) is unaffected by endianness, which is important for a cross-platform application like BitTorrent. It is also fairly flexible, as long as applications ignore unexpected dictionary keys, so that new ones can be added without creating incompatibilities.

Contents

1 Encoding algorithm
2 Features & defects
3 See also
4 References
5 External links

Encoding algorithm [edit]

Bencode uses ASCII characters as delimiters and digits.

- An integer is encoded as **i**<*integer encoded in base ten ASCII*>**e**. Leading zeros are not allowed (although the number zero is still represented as "0"). Negative values are encoded by prefixing the number with a minus sign. The number 42 would thus be encoded as `i42e`, 0 as `i0e`, and -42 as `i-42e`. Negative zero is not permitted.
- A byte string (a sequence of bytes, not necessarily characters) is encoded as <*length*>**:**<*contents*>. The length is encoded in base 10, like integers, but must be non-negative (zero is allowed); the contents are just the bytes that make up the string. The string "spam" would be encoded as `4:spam`. The specification does not deal with encoding of characters outside the ASCII set; to mitigate this, some BitTorrent applications explicitly communicate the encoding (most commonly UTF-8) in various non-standard ways. This is identical to how netstrings work, except that netstrings additionally append a comma suffix after the byte sequence.
- A list of values is encoded as **l**<*contents*>**e** . The contents consist of the bencoded elements of the list, in order, concatenated. A list consisting of the string "spam" and the number 42 would be encoded as: `l4:spami42ee`. Note the absence of separators between elements.
- A dictionary is encoded as **d**<*contents*>**e**. The elements of the dictionary are encoded each key immediately followed by its value. All keys must be byte strings and must appear in lexicographical order. A dictionary that associates the values 42 and "spam" with the keys "foo" and "bar", respectively, would be encoded as follows: `d3:bar4:spam3:fooi42ee`. (This might be easier to read by inserting some spaces: `d 3:bar 4:spam 3:foo i42e e`.)

There are no restrictions on what kind of values may be stored in lists and dictionaries; they may (and usually do) contain other lists and dictionaries. This allows for arbitrarily complex data structures to be encoded.

Features & defects [edit]

Bencode is a very specialized kind of Binary coding with some unique properties

- For each possible (complex) value, there is only a single valid bencoding; i.e. there is a bijection between values and their encodings. This has the advantage that applications may compare bencoded values by comparing their

encoded forms, eliminating the need to decode the values.

- Many BE codegroups can be decoded manually. Since the bencoded values often contain binary data, decoding may become quite complex. BEncode is not considered a human-readable encoding format.
- Bencoding serves similar purposes as data languages like JSON and YAML, allowing complex yet loosely structured data to be stored in a platform independent way.

However, this uniqueness can cause some problems

- There are very few Bencode editors[1]
- Because Bencoded files contain binary data, and because of some of the intricacies involved in the way binary strings are typically stored, it is often not safe to edit Bencode files in text editors.

See also [edit]

- BitTorrent

References [edit]

1. ^ "BEncode Editor". µTorrent Community Forums. Retrieved 21 August 2013.

External links [edit]

- Bencoding specification
- File_Bittorrent2 - Another PHP Bencode/decode implementation
- The original BitTorrent implementation in Python as standalone package
- BEncode Editor
- bencode-tools - a C library for manipulating bencoded data and a XML schema like validator for bencode messages in Python
- Bencode parsing in Java
- Bencode parsing in C
- There are numerous Perl implementations on CPAN

Categories: BitTorrent | Data serialization formats

This page was last modified on 23 March 2014 at 18:32.

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.

Privacy policy About Wikipedia Disclaimers Contact Wikipedia Developers Mobile view